Leonid Shuster

ECE 167

2/29/20

Lab Report 3

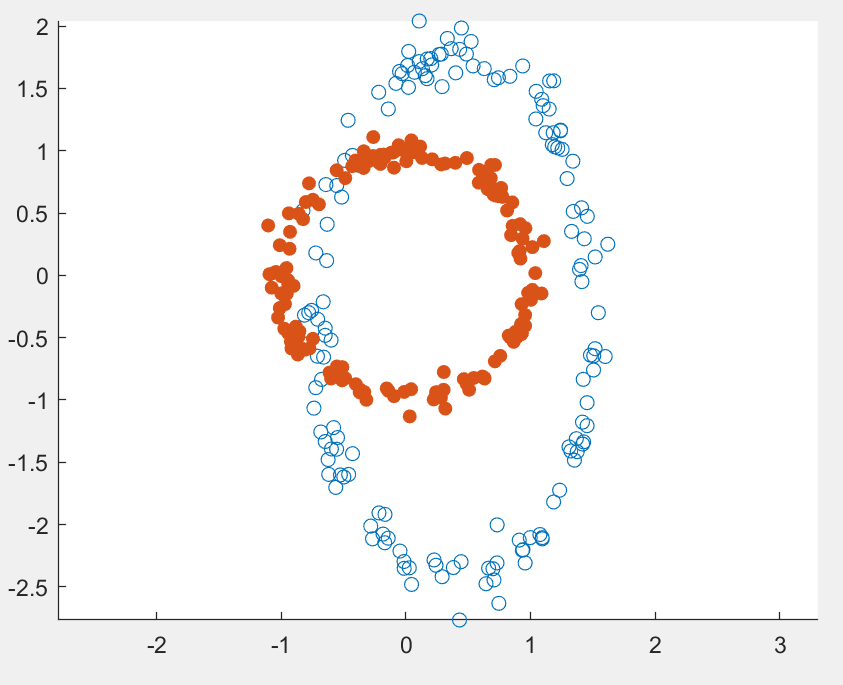
**Introduction**

In this lab, we were tasked with learning how to calibrate an IMU sensor. In the first part, we looked at a hypothetical 2D sensor and removed the noise from its measurements. In the second part, we took accelerometer and magnetometer measurements with the IMU sensor and solved for its null shifts and offsets on each axis. In the third part, we took gyroscope measurements with the IMU sensor over an hour and plotted the bias drift. In the fourth part, we rotated the sensor 180 degrees along each axis and took gyroscope measurements and plotted it to see how close to the correct angle it was measuring. In the fifth part, we used simulated accelerometer and magnetometer tumble data of the sensor and applied various calibration techniques to correct the data. And in the sixth part, we took actual accelerometer and magnetometer tumble data with the sensor and applied those same calibration techniques.

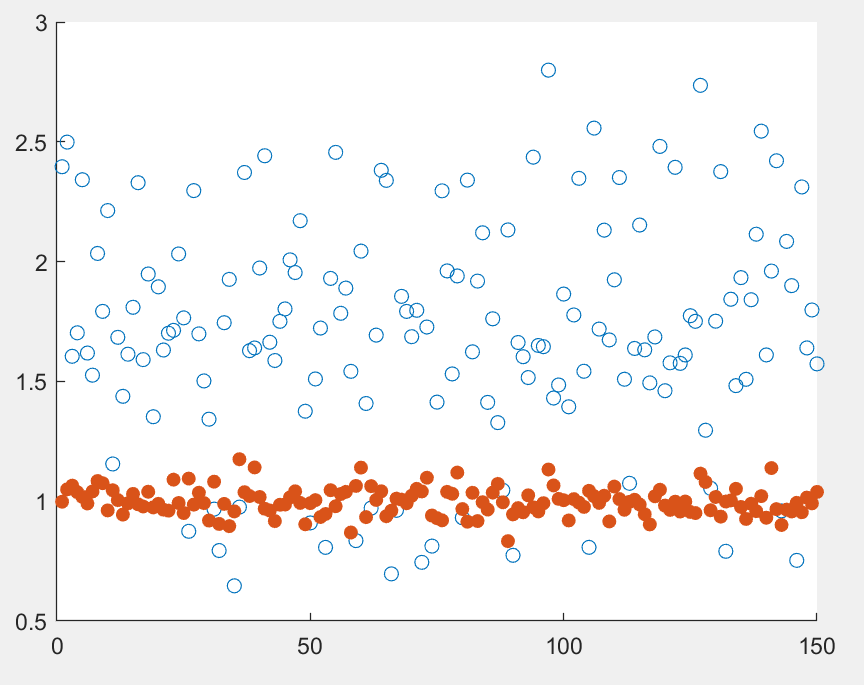
**Part 1: Ellipsoid Calibration using Simulated Data**

In the first part of the lab, we were told to remove the noise from the measurements of a hypothetical 2D sensor. An ideal 2D sensor, when measuring the x and y components of some fixed vector, should result in a circle of radius equal to the length of the vector. However, when noise is introduced, null shift and scale factors change the shape of the circle into an ellipse. Our task was, given the points on the ellipse, to find the scale factor and null shifts and reconstruct the original circle. In order to do so, I started with the equation of an ellipse,

where xo, yo, a, and b were the null shifts and scale factors on the circle we were trying to find and x and y were points on the ellipse. I then expanded the equation and solved for x2 in terms of everything else, put the constants of every term (x, y) in an nx4 matrix and the variables of every term (xo, yo, a, b) in another 4x1 matrix. Then, using least squares, I was able to solve for the 4x1 matrix values, and then for the xo, yo, a, b values. I found that xo = 0.7923, yo = 0.2658, a = -0.1801, and b = 1.0349. Then, using the equation of a circle, where and with the values previously stated, I was able to accurately recreate the circle (see Figure 1).



I then plotted the 2-norm of the pre-calibrated and post-calibrated data to see how they compared (see Figure 2).



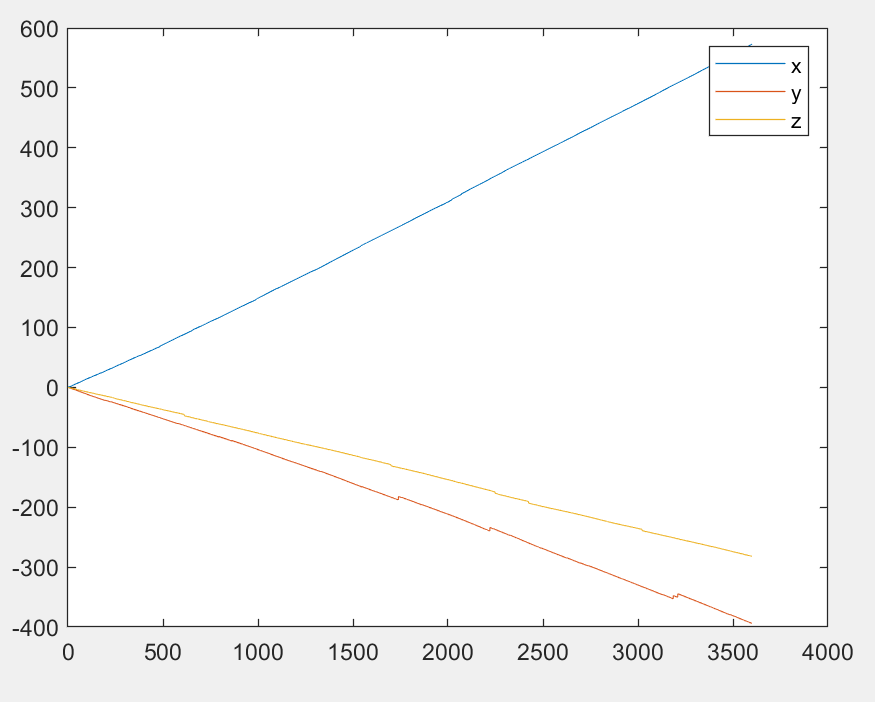
**Part 2: Naive Calibration of Magnetometer and Accelerometer**

In the next part of the lab, we were told to take accelerometer and magnetometer measurements with the IMU sensor and solve for the null shift and offset on every axis. First, I took accelerometer measurements on every axis of the sensor including 180 degrees across it. In my code, I read from every axis at 50 Hz using a timer and printed them out to the serial port. Then, to find the scale factor and null shift along each axis, I created an nx2 matrix for every axis with the measurements I got for them (concatenated with a column of ones), which would then be multiplied by a 2x1 matrix with the scale factor and null shift to get an nx1 matrix of the expected value of gravity, either 9.80665 for the positive axis or -9.80665 for the negative one. Using least squares, I solved for the 2x1 matrix. For the scale factor and null shift of the x-axis, I got 0.0006 and -0.2613, for the y-axis I got 0.0006 and -0.1297, and for the z-axis I got 0.0006 and -0.0025.

I followed a similar procedure for the magnetometer measurements. After finding the point of horizontal intensity, I took measurements along every axis, including 180 degrees along it. I then once again put the values into a nx2 matrix which would then be multiplied by a 2x1 matrix with the scale factor and null shift to now get an nx1 matrix of the expected value of the magnetic field, either 0.4784 for the positive axis or -0.4784 for the negative one. For the scale factor and null shift of the x-axis, I got 0.0054 and -1.9114, for the y-axis I got -0.0027 and 0.5226, and for the z-axis I got -0.0036 and -0.1272.

**Part 3: Gyro Bias and Bias Drift**

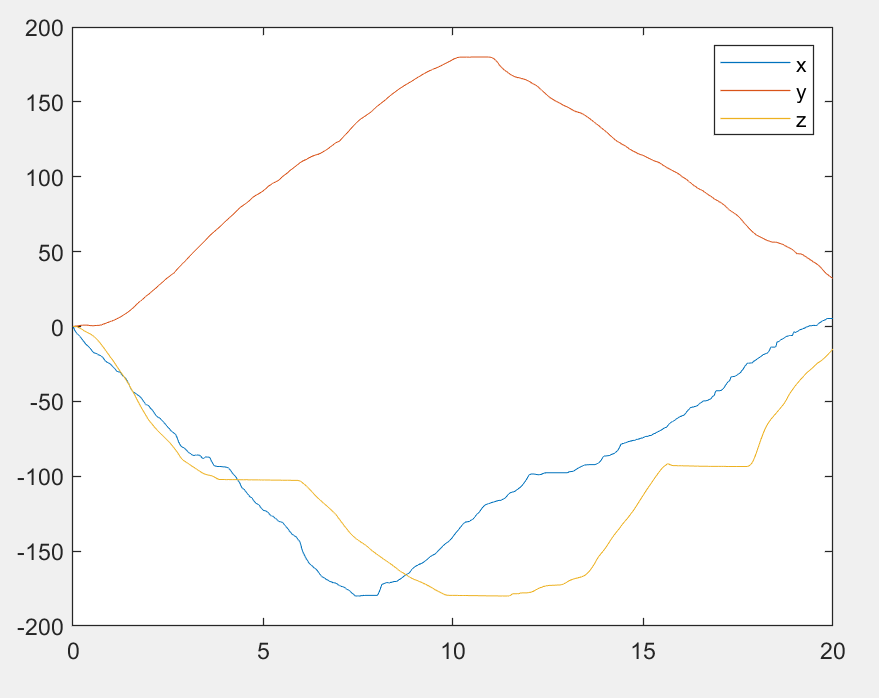
In the next part of the lab, we were tasked with finding the gyroscope bias drift of our sensor. The first thing we had to do was get an estimate of the bias for every axis, which I did by taking measurements along every axis for 10 seconds and averaging them. Then, I set up the sensor to take measurements for every axis for an hour. After doing so, I subtracted the initial bias from all the measurements and integrated over time and plotted it to see what the drift in degrees per hour was (see Figure 3).



I found that my bias drift was much larger than I had expected.

**Part 4: Gyro Scale Factor via Angle Integration**

In the next part of the lab, we were tasked with rotating the sensor and seeing how close we could get its measurements to the actual angle it was measuring. Like the previous part, I took an average of 10 seconds of data along each of its axes to get an estimate of the sensor’s bias. Then, I took measurements of the sensor as I rotated it 180 degrees and back along each of its axes. After doing so, I once again subtracted the initial bias from all the measurements, integrated over time, scaled it, and plotted it to see if the sensor was measuring the correct angle (see Figure 4).

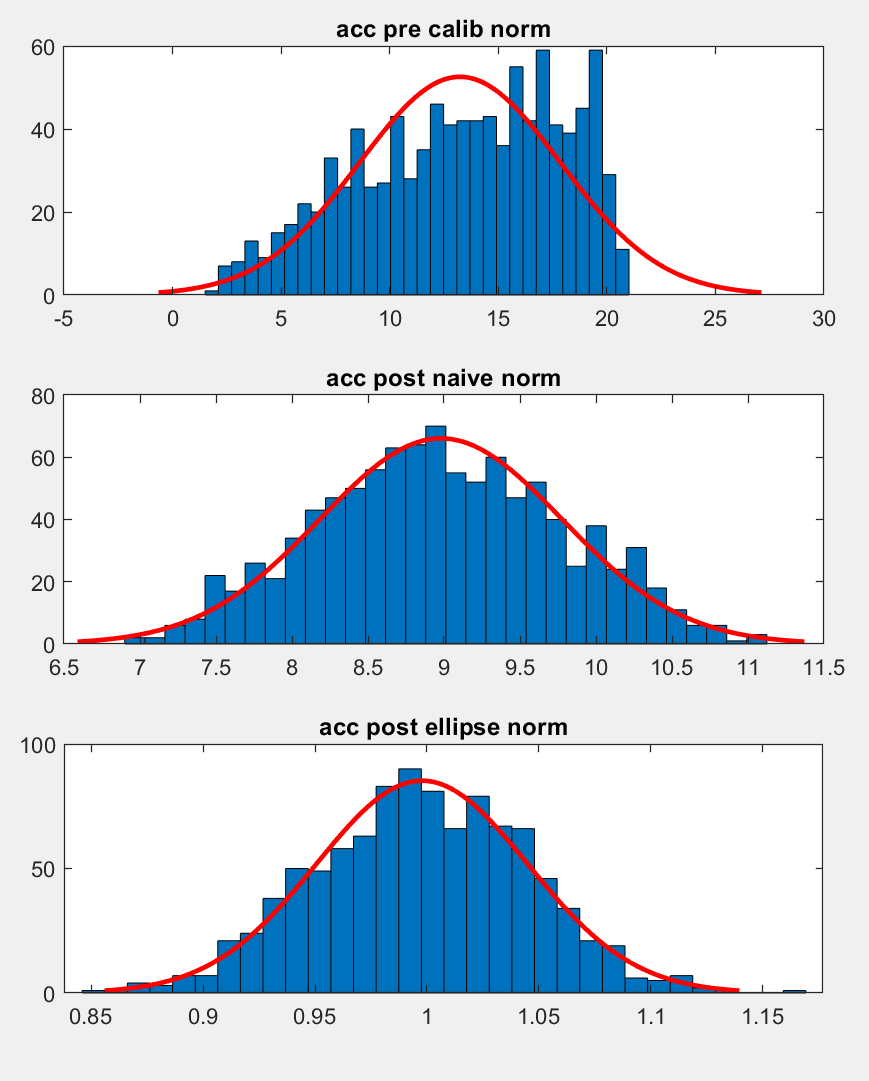
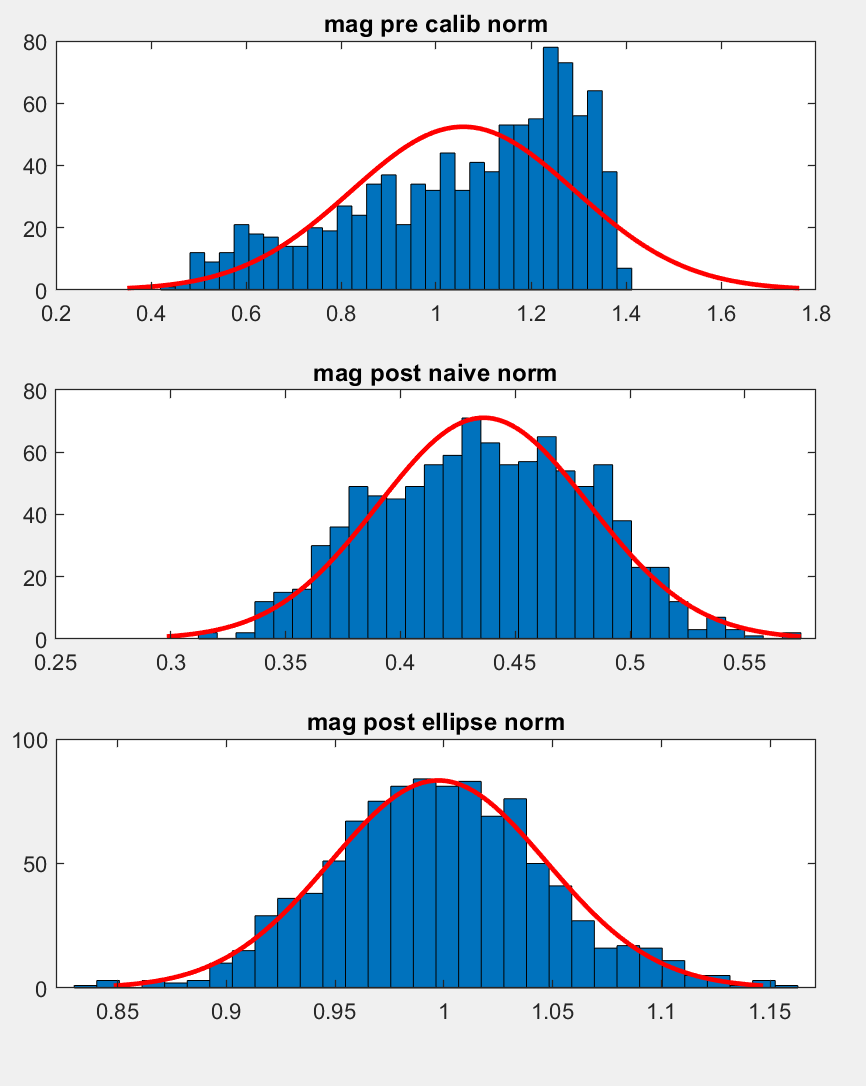


I found that the sensor pretty accurately found the correct angle with several bumps along the way.

**Part 5: Tumble Test Simulation**

In the next part of the lab, we were tasked with using various calibration techniques on simulated accelerometer and magnetometer tumble data from the sensor. The first technique I used was naive calibration, which I did by calculating the scale factor and null shift from the data simulated. I found the scale factor by dividing the range of the expected value (2\*9.80665 for the accelerometer or 2\*0.4594 for the magnetometer) by the simulated data range, and the null shift by taking the average of the data. I then subtracted the null shift and multiplied the scale factor by the data simulated. For the next calibration technique, I first scaled the data simulated into units of g’s and Earth’s magnetic field. With this scaled data, I then used the MatLab functions provided to calibrate and correct the data.

After using the two techniques, I then plotted the norm of both (see Figures 5 and 6).

I also calculated the means and standard deviations (see Tables 1 and 2).

Accelerometer

|  | Mean | Standard Deviation |
| --- | --- | --- |
| No Calibration | 19.9828 | 4.9763 |
| Naive Calibration | 9.0001 | 0.7367 |
| Ellipsoidal Calibration | 0.9975 | 0.0498 |

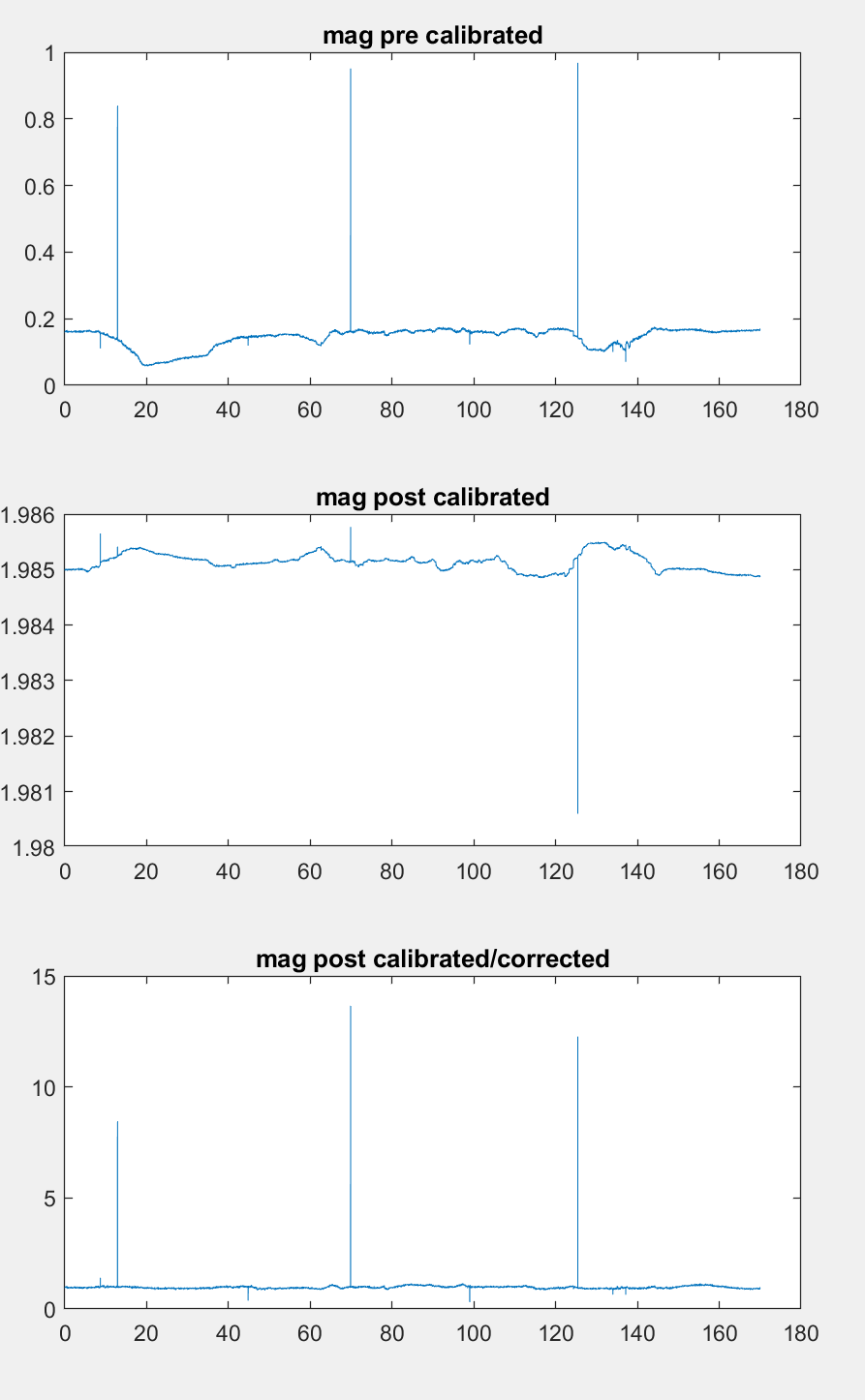
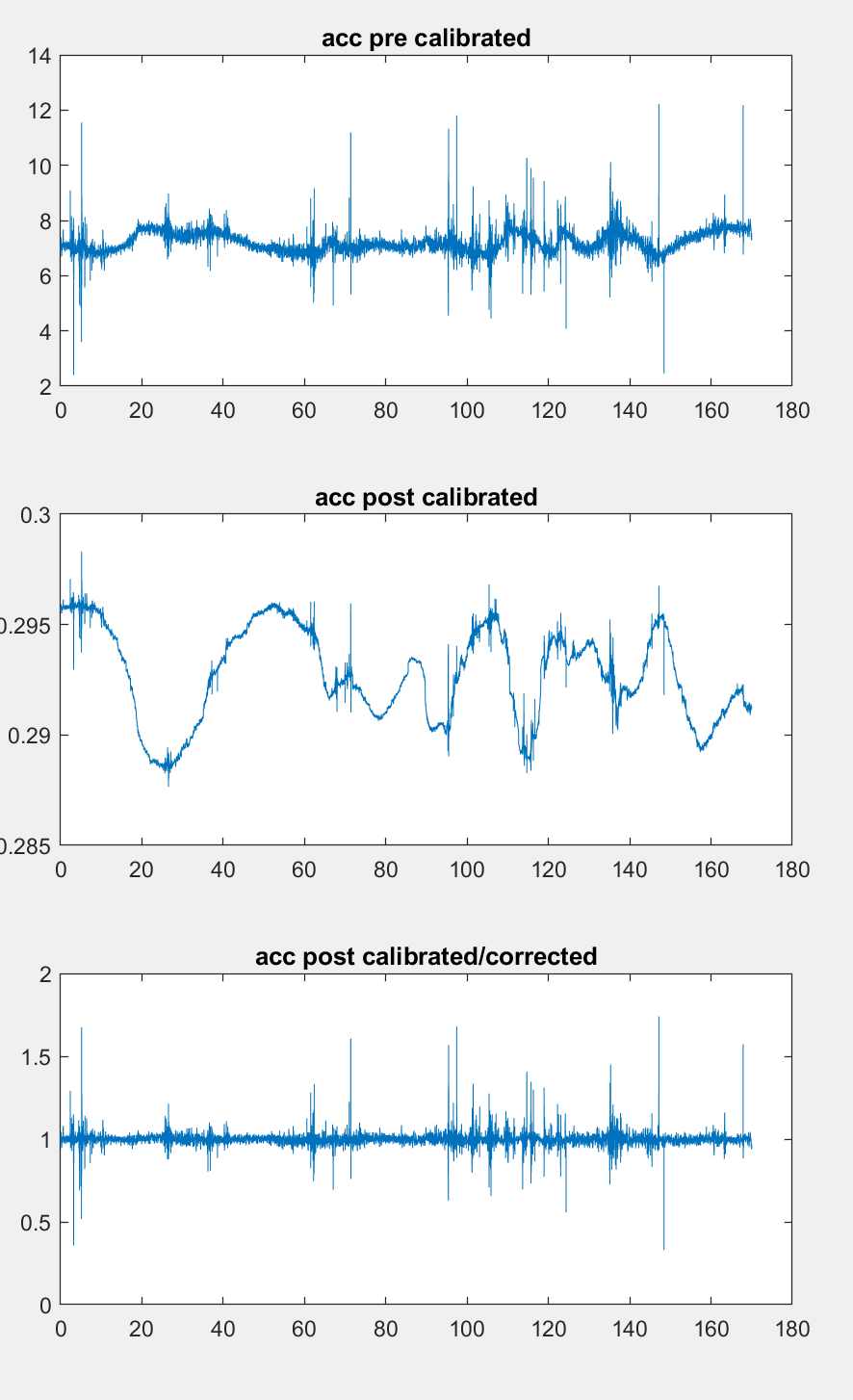
Magnetometer

|  | Mean | Standard Deviation |
| --- | --- | --- |
| No Calibration | 16.3223 | 5.1293 |
| Naive Calibration | 0.4437 | 0.0341 |
| Ellipsoidal Calibration | 0.9976 | 0.0485 |

I found that while the naive calibration had closely calibrated the data, the calibration technique provided from the MatLab functions much more accurately corrected the data.

**Part 6: Tumble Test for Accelerometer and Magnetometer**

In the final part of the lab, we were tasked with using the same calibration techniques but on actual measured accelerometer and magnetometer data. After taking measurements for 2 minutes, I then used the various calibration techniques. For the first technique, I used the scale factor and null shift I found in Part 2. For the next technique, I once again used calibration and correction MatLab functions provided. I calculated the norms for both, and plotted the norms vs. time (see Figures 7 and 8).



I also calculated the means and standard deviations (see Tables 3 and 4).

Accelerometer

|  | Mean | Standard Deviation |
| --- | --- | --- |
| No Calibration | 7.2135 | 0.3859 |
| Part 4 Calibration | 0.2927 | 0.0021 |
| Ellipsoidal Calibration | 0.9988 | 0.0352 |

Magnetometer

|  | Mean | Standard Deviation |
| --- | --- | --- |
| No Calibration | 0.1443 | 0.0307 |
| Part 4 Calibration | 1.9851 | 1.5631e-04 |
| Ellipsoidal Calibration | 0.9765 | 0.1514 |

I found that the means and standard deviations from the calibration using the scale factor and null shift from Part 4 were not as precise as the technique using the MatLab functions, which I believe is due to my sensor drifting over time.

**Conclusion**

After having gone through the lab, I feel like I fundamentally understand how to calibrate an IMU sensor. If I were to do this lab again, I would do many test runs with the calibration I used to make sure that they correctly calibrated the data.